Background Problem Set

This problem set asks questions that we would expect students with an undergraduate preparation for algorithms to be able to answer; most were on a not so recent but legendary final exam. The ideas are either so fundamental you need to know them, or we will expand/rely on them in this class. You may use the book Algorithms by DasGupta, Vazirani and Papadimitriou to fill in your background. Indeed, we will cover some of that material, particularly Chapter 7, in this course. Please refrain from searching the web beyond that book for this homework and try to do this on your own or with help from the course staff via Piazza or office hours. There is an online version of the book that can be found using google; search for "Vazirani Algorithms chap0.pdf", and then well, you figure it out.

Due Date: Friday January 27. Submissions will be electronic, through gradescope.

1. Some short answers.

(a) What is an asymptotically tight upper bound on the recurrence
$T_1(n) = 4T_1(n/3) + O(n^2), T_1(1) = O(1)$?
And for the recurrence $T_2(n) = 27T_2(n/3) + O(n^2), T_2(1) = O(1)$?

(b) Eureka! A woman named Nessarts figured out how to multiply 4×4 matrices using 32 scalar multiplications and 67 additions and subtractions. She finds that the elements of the matrix don't have to be scalars — her method also works on large matrices split into 16 blocks (4×4). How fast can she multiply $n \times n$ matrices? (You can assume n is a power of a convenient number and use asymptotic notation.)

(c) In a max-flow problem, we have found a minimum cut and it contains an edge e. Prove or give a counterexample: If we increase the capacity of e, the value of the max flow will increase.

(d) Express the problem of maximizing $\min(x+y, y+w, 3x+w)$ where $x+y+w = 1$ as a linear program.

(e) In the vertex cover problem, the input is an undirected graph $G = (V, E)$ and the goal is to find the smallest set $S \subseteq V$ of vertices such that every edge touches a vertex in S. Here as an integer linear program for this problem:

$$\min \sum_v x_v$$

$$\forall e = (u, v) \in E, x_u + x_v \geq 1.$$

$$\forall v \in V, x_v \in \{0, 1\} \qquad (*)$$

Describe how to use a solution to the linear program, with the integrality constraints (*) removed, to provide a 2-approximation for the vertex cover problem on a general (non-bipartite) graph. (You are not allowed to look at the graph, only the values of x_u in your algorithm.)

2. Repairing a Minimum Spanning Tree

Suppose you are given a graph $G = (V, E)$ and a minimum spanning tree $T \subseteq E$ of G. An evildoer deletes a node $v^* \in V$ and all edges that connect to v^*: the new graph without v^* is called $G' = (V', E')$. Assume G' is connected.

Show how to use T to find a minimum spanning tree T' of G' in time $O(|E| + d^2 \log d)$, where d is the degree of v^* in the original spanning tree T. Why is your algorithm correct?

3. Consider a set of points $S = \{p_1, \ldots, p_n\}$ where $p_i = (x_i, y_i) \in \mathcal{R}^2$. A point p_j is said to *dominate* another point p_k if (a) $x_j \geq x_k$ and (b) $y_j \geq y_k$.

(a) Give an efficient algorithm that given a set of points S finds all points of the set S which *do not* dominate any other point in S.

(b) Give an efficient algorithm to find the largest subset of points U where for each pair of points in U neither dominates the other. (Hint: Dynamic Programming.)

4. The coloring problem is, given a graph $G = (V, E)$ and a number k, to find a way to use color the vertices with k colors so that for each edge $e = (u, v) \in E$ has u and v being a different color. We will reduce 3-SAT to this problem. We make a node for each of the n variables v_i and its negation v_i' and connect them by an edge. One will be colored with one of many "true" colors and the other will be colored "false". We then make a clique on vertices $w_1, \ldots, w_n$. (Note, this forces each vertex, w_i, to be a different color.) We connect each node w_i to every variable node and its negation *except for* v_i and v_i'. Thus, the color of w_i can be used to color one of v_i or v_i' true and the other of the variable vertices could be colored false (an $n + 1$st color.) See figure 1 for a sketch of a 3-variable gadget.

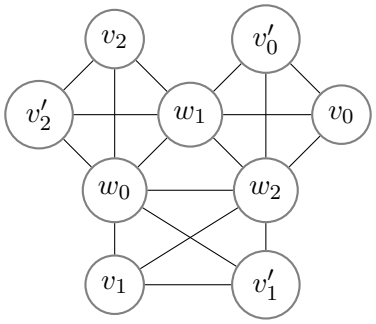

Figure 1: Three variable gadget.

We will introduce a vertex for each clause. We wish to be able to color the clauses with one of the n true colors if and only if there is a satisfying assignment. You need to figure out how the clause vertex should be connected. For example, what should the vertex for a clause consisting of $(x_i \vee x_j \vee \overline{x_k})$ be connected to? What is the general rule? Be concise.

5. We are given a bit pattern s_1 of length m and a bit pattern s_2 of length n, where $m < n$. We wish to find the location of any occurrence of the pattern s_1 within s_2 with k or fewer mismatched bits.

(a) Give an $O(nm)$ algorithm for this problem.

(b) Give an $O(n \log n)$ time algorithm for any k. (Hint: Represent the bits as +1 or -1. What's a fast way to compute convolutions?) (You may use the computation model where multiplications of real numbers require $O(1)$ time assuming reasonable sized numbers.)

6. Let $\pi = \pi_1 \pi_2 \ldots \pi_n$ be a permutation of $1\ 2 \ldots n$. A reversal $\rho(i, j)$ reverses the contiguous subsequence between i and j i.e. transforms

$$\pi_1 \ldots \pi_{i-1} \pi_i \pi_{i+1} \ldots \pi_{j-1} \pi_j \pi_{j+1} \ldots \pi_n$$

into

$$\pi_1 \ldots \pi_{i-1} \pi_j \pi_{j-1} \ldots \pi_{i+1} \pi_i \pi_{j+1} \ldots \pi_n$$

The *sorting by reversals* problem is to find the minimum number of reversals that transforms a given permutation to the identity permutation. For example, the permutation [3,4,2,1] can be sorted by flipping [3,4] to get [4,3,2,1] and then flipping this entire sequence.

(a) Let's extend the permutation by adding $\pi_0 = 0$ and $\pi_{n+1} = n + 1$ to the ends (π_0 and π_{n+1} are not moved during sorting). We say there is a breakpoint between π_i and π_{i+1} if they are not consecutive (in either order). For example, the permutation [3,4,2,1] is converted to [0,3,4,2,1,5] and there are three breakpoints: between (0,3), (4,2), and (1,5). Note there are not breakpoints between (3,4) or between (2,1) as each adjacent pair consists of consecutive elements.

Give a lower bound on the number of reversals needed to sort a permutation, π, if π has $b(\pi)$ breakpoints.

(b) Give an algorithm for the sorting by reversals problem with approximation ratio of at most 4 with respect to the minimal number of necessary reversals. (Hint: Any maximal run of elements with no breakpoints should be kept together.)

7. Suppose there are n experts. Each day, each one announces whether it will rain or shine. There is one expert who is perfect.

Each day, you will predict rain or shine; apparently using the expert's advice. You make a mistake on a day if you choose the wrong thing.

Devise a strategy for choosing rain or shine each day and give an upper bound on how many mistakes you make. Note: the process continues forever.